Scale-space theory applied to text analysis

Nathanael Aff
naff@mail.sfsu.edu

January 16, 2015

Introduction

The way in which language is represented is fundamental to techniques used throughout the field of natural language processing and text.mining. The bag-of-words(BOW) model is the most commonly used model for tasks such as text classification and information retrieval. This model represents text as vector of single-word, unigram, frequencies and is used, for instance, in constructing a vector space model of documents. In this model a document is represented by a weighted vector of term frequencies and similarities between two documents or a query and a document can be determined using some simlarity metric on the vector space, cosine similarity, for example. This language model can be expanded to consider some window of adjacent words as a feature. However, the number of adjacent words, termed an ngram, is generally fixed before hand. More complex models designed to extract richer sets of information such as entity relationships may add features such as part-of-speech tagging and syntactic parsing. However, the underlying BOW language model determines features used in most text mining processes.

A drawback of the BOW model is that contextual information is lost when considering only word frequencies. Some of this information is regained through additional processing but at a computational cost. To overcome this drawback of the BOW model the scale-space representation of text was proposed by Shuang Yang. [9] In this model, text is represented as a 2-Dimensional signal, $f(x,y)$, with x representing the time domain and y the semantic domain. This signal is then embedded in a scale-space parameterized by a scale variable, σ. At the finest scale with $f(x,y)$ is simply the original binary signal, where each word is represented as an element from the vocabulary at some position.. As the scale parameter of some smoothing kernel is increased, local features are lost, leaving only large scale features. The scale-space model was largely developed in the area of digital image processing. With an image, the natural resolution of an image may not be known and progressively smoothing local features allows for the automated discovery of more global patterns.

The motivation for applying the scale-space model to a 2D text is to efficiently discover features of text that form at different scales, from local word relations to document scale features. Additionaly, the 2D representation may allow for many well-developed signal and image processing techniques to be adapted to text analysis.

Related Work

There have been other attempts to incorporate positional information into other language models. Lafferty and Blei have modeled topics over time but their model also uses changes in the multinomial distribution based on a bag-of-words model. [1] This model is also aimed at modeling language change at the document level over time. Lebanon et al. developed a method for analyzing within-document changes but similar to Blei and Lafferty the model uses changes in word frequency over time based on the BOW model. [3] This model takes a geometric view of a document – the manifold created by the multinomial simplex formed by the term-distribution is used to determine changes of meaning or topic within the document. Other techniques incorporate proximity information. The closest model to Yang's is the that of Lv and Zhai, which they term a positional model. Locations of each word is smoothed over a proximity

graph, much like in the scale-space model. Although Lv and Zhai's terminology and approach is different the underlying model may be very similar to Yang's although Lv and Zhai have not related their methods to signal or image processing techniques. A further comparison of their methods to Yang's might be fruitful.

Scale Space Filtering

The formulation of scale-space filtering arose in image processing as a means of identifying the appropriate scale of analysis for any 2-dimensional signal. The requirements for an appropriate scale-space convolution kernel have been formalized in different ways but the basic requirements are described by Lindeberg. [5] They are: shift invariance, linearity and the non-creation of new extrema for 1-dimensional signals. For higher dimensional signals, this property doesn't hold but is expressed more generally as the non-enhancement of local extrema for signals of higher dimension. The scale-space kernels can also be found as the solution to the isotropic diffusion (heat) equation. [5]

The 2D Gaussian kernel is defined:

$$\ell(x_1, x_2, \sigma) = \frac{1}{\sqrt{2\pi\sigma}} \exp(\frac{-(x_1^2 + x_2^2)}{2\sigma})$$

In the discrete case, the The kernel has additional useful properties when used for filtering : it is a separable kernel, which allows for efficient computations but more importantly, will allow for separately smoothing the semantic and spatial domain in our application. The Gaussian kernel is also infinitely differentiable and the derivative of a Gaussian is a Gaussian, so similar properties will hold for filtering with the derivatives of the kernel. Although Lindeberg has also presented a kernel for smoothing in the discrete case, most common implementations use a sampled Gaussian in place of a discrete kernel. [4]

Application to Text

We follow Yang's application of scale-space filtering to text. A document is represented as a signal $f(x,y)$. A vocabulary V is simply a set of unique words, $\{w_1, w_2, ..w_n\}$ presented below on the vertical axis. Positions are simply the ordered positions of the words within the text, represented here on the x-axis. Filtering in the spatial domain is carried out by convolution with the sampled gaussian :

$$\ell(x; \sigma) = \frac{1}{\sqrt{2\pi\sigma}} \exp(\frac{-x^2}{2\sigma})$$

Filtering in the semantic domain, however, is not as straightfoward. The general approach is to define a similarity measure between words in the vocabulary and use a filter that smooths words based on this metric. A Gaussian kernel is used but for each word, z, related to word y smoothing takes place over the measured dissimilarity between z and y, d_{yz} with the kernel:

$$\ell(y, z; \sigma) = \frac{1}{\sqrt{2\pi\sigma}} \exp(\frac{d_{yz}^2}{2\sigma})$$

In order to carry out this smoothing, a proximity matrix is built based on pointwise-mutual information(PMI). [2] The measure is a simple measure of co-occurence:

$$\text{pmi}(x,y) = \log \frac{p(x,y)}{p(x)p(y)}$$

PMI has shown to be competitve with more computationally complex measures when calculated on large data sets. [7]

The proximity matrix constructed in this way can also be viewed as a semantic graph over which filtering takes place. Mei et al. proposed an optimization framework for smoothing over a semantic (or other similarity) graph. [6] In their formulation a word-word proximity graph is smoothed over by minimizing the objective function:

$$O(C) = (1-\lambda) \sum_{y \in V} w(y)(f_y - \tilde{f}_y)^2 + \lambda \sum_{(y,z) \in E} w(y,z)(f_y - f_z)^2$$

for a graph $G = \langle V, E \rangle$, where:

- $y, z \in V$ are words or vertices of the graph
- $(y,z) \in E$ is a weighted edge between words.
- f_y is the smoothed value of y in the graph
- $\tilde{f}_y$ is the smoothed value of y
- $w(y)$ is the weight of y in G
- $w(y,z)$ is the weight of edge (y,z)

The degree of the node is used as the importance weight:

$$w(y) = Deg(y) = \sum_{y \in V} w(y,z)$$

To calculate a smoothed value at a single node, the first order partial derivative with respect to f_y is set to zero and solved. Substituting $Deg(y)$ gives us [6] :

$$f_y = (1-\lambda)\tilde{f}_y + \lambda \sum_{z \in V} \frac{w(y,z)}{Deg(y)} f_z$$

Intuitively, this is a weighted smoothing of $\tilde{f}_y$ based on some weighting of the node and it's neighbors. Interpreting the arc weights, $w(y,z)$, as a distance from a node y, this can be viewed as the the convolution of f_y with some kernel K [9]:

$$f_y = \sum_{z \in V} \tilde{f}_y \cdot K_{yz}$$

Since the Gaussian kernel is separable, we can filter in the x then y domain. The convolution for the textual signal $f(x,y)$ can be expressed:

$$\ell(x,y,\sigma_x,\sigma_y) = \ell(x,\sigma_x)\ell(y,\sigma_y)$$

Implementation Overview

In implementing the procedures outlined by Yang an attempt was made to replicate as closely as possible the methods he described. Many of the techniques described are standard either in the fields of text analysis or signal processing. The implementation was written in Python and where available standard libraries were used. For convolution and filtering, methods from the Scipy library were used and for natural language processing the NLTK library was used.

Despite this, several key procedures described by Yang left room for interpretation. Most importantly, some details regarding the building of the semantic graph using PMI scores and smoothing over this graph were not fully explained. In our implementation, the PMI scores were built using the NLTK function using a window of two – that is, only adjacent words (after stopwords were removed) were included.Using a larger window would likely result in significant changes to the PMI scores and the semantic graph.

PMI scores for low frequency words are unstable – infrequent word combination can rank very high. For our implementation we only considered bigrams with a fequency greater than two but taking bigrams with a count greater than three is more standard. More were included in this context since we were building the graph from a fairly small corpus of 1,000,000 words. It has been shown that effectiveness of PMI scores as a measure of association is greatly effected by the size of the corpus used to build the measure. [7] Re-creating the semantic graph with a larger corpus might improve the results.

In addition to parameters used in building the semantic graph, an adjustment is required to make the raw PMI scores usable as a distance metric. Yang did not describe is method for creating a distance metric. We constrained the PMI to a 0-4 range by normalizing to 0-1 and scaling, then the additive inverse of the PMI was used a the distance measure. However, the outlier bigrams with very large PMI scores lead to bigrams with moderate scores being bunched in a fairly small range. This greatly affects the results for smoothing over the semantic graph. The optimization framework for smoothing over the graph allows for weighting of the importance of a word in relation to its neighbors and weight of its node within the graph. [6]

In a semantic graph using PMI scores the weighting of a word node— the number of words it is connected to — partially taken into account in the PMI score. Since the denominator of the PMI score is $p(x)p(y)$ more frequent words are ranked lower. However, in constraing the distance based on PMI scores, the bunching of moderately sized scores led to relatively similar smoothing over bigrams that should probably be more distinct. This led to generally weak and undifferentiated smoothing over bigrams. For example, in the keyword method described below, the first step is to smooth over the semantic level at some fixed sigma. When sigma was increased, common words took on too much weight. Their PMI scores should be weaker since they occur more frequently but the relatively undifferentiated PMI distance metric led to common words being weighted more heavily compared to less frequent words with possibly stronger associations. This effect would need to be corrected for methods using smoothing over

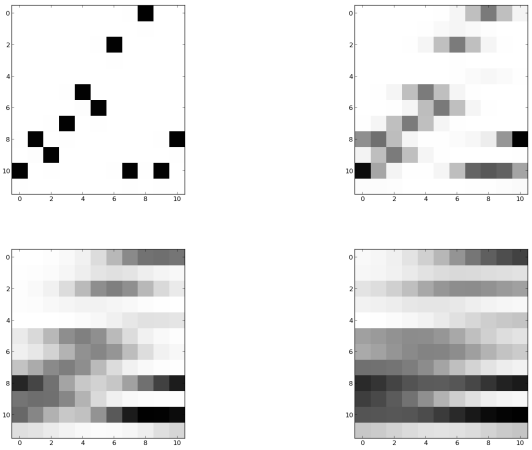

Figure 1: Smoothing over the semantic and spatial domain of a the sentence 'New York Times offers free phone as gift for new customers'. Results were normalized to a 0-1 range so smoothing is more even then the plots appear to show.

the semantic graph to be effective.

Test results

As mentioned above, the implementation was programmed in Python using the Natural Language ToolKit(NLTK) and Scipy signal processing libraries. The corpus used to build the semantic graph was taken from a section of the New York Times Giga Corpus. The text was cleaned and tokenized but was not stemmed. This approach might be more warranted if a larger corpus was used for calculating bigram association. The semantic graph was based on the constrained PMI measures.

Since the Gaussian kernel is separable, smoothing over the semantic and spatial domain were calculated in two steps. Smoothing over the semantic domain was implemented by calculating the sampled Gaussian kernel for each word related to a given word-node based on the constrained PMI distance from the word-node.

Figure 1 illustrates a sample smoothing over a short sentence using a limited vocabulary. Since the semantic domain is large — the semantic graph containted around 20,000 unique words — a small sample text with a reduced vocabulary was used in order produce a sample visualization. The sentence used is based on the example in Yang's paper so at least a rough comparison could be made to his results.

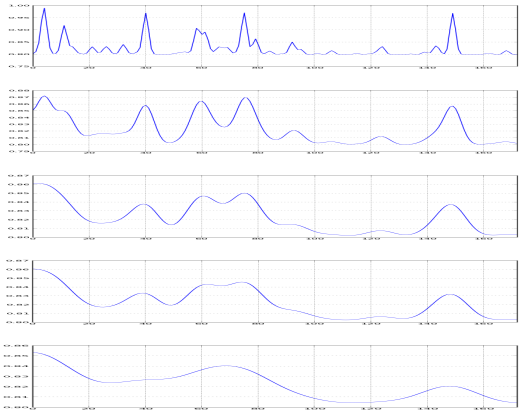

Figure 2: Progressive smoothing in the spatial domain after a fixed smoothing in the semantic domain.

We used the sentence "New York Times offers free phone as gifts for new customers in New York". A vocubalary was used consisting of 'new', 'york', 'time','offers', 'gift', 'free', 'phone', 'shop','people' 'customer', 'service', 'support' ,'city', where the additional words in the vocabulary were selected for their high PMI score relative to words in the sentence.

The degree of smoothing is greater than represented in images of smoothing for sigma =2 and sigma=3 due to the plotting program normalizing the values to a 0-1 range. While it is subtle, the smoothing in the semantic domain is weaker than in the spatial domain. This is likely due to the weakly differentiated distance metric that were bunched in a middle range, as mentioned in the implmentation section. Although smoothing in the two domains can be carried out with different sigma values to achieve a better balance, overall, a better distance metric would be needed to achieve more meaningful filtering.

Keywords

Again following Yang, a simple keyword extraction method was implemented. The semantic domain was filtered at a fixed sigma ($\sigma_y = C$). A low filter window was used since higher filtering led to over-weighting common words — for example, on one text 'one' and 'white' were the highest scored keywords. A vector of values summed over the semantic domain was then used to extract keywords. This process is shown in figure 2, where the top image is the marginal values of the semantic domain after a filtering at

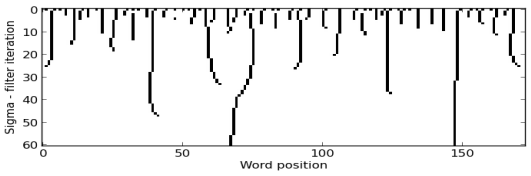

Figure 3: Relative maxima traced from coarse to fine smoothing.Greater smoothing is lower on the graph. The curve in the maxima roughly show the direction in which smaller maxima merged with larger.

sigma = 1.

Keywords were extracted by tracing relative maxima from a coarse smoothing back to a their origins (figure 3). A draft of this paper was used to extract keywords. Keywords extracted from the top level down were: sigma, lebanon, techniques, node, graph $\rightarrow$ neighbors, natural $\rightarrow$ hand, equation $\rightarrow$ due $\rightarrow$ measure, general $\rightarrow$ metric, frequencies, significant, scores.

For comparison, a list of the top ten words by frequency are: model, used, semantic, pmi, smoothing, graph, scale, kernel, word, words. Ignoring stopwords, the most common bigrams were pmi scores, scale space, semantic graph, adjacent words, optimization framework. The top bigrams in this case seem to catch the sense of the text better than the keyword method.

Future Development

Although the results for the keyword were not satisfying, there are aspects to the approach that could be improved and could possibly yield richer information than ngram frequencies. For example, the 'ridges' tended to form around word clusters but keywords using the previous method only corresponded to the single word which was, roughly, the origin point for the local maxima. More generally, these keywords could simply be extracted after the fixed smoothing in the sematic domain. Since we have preserved the spatial structure of the text this information should be able to be used in capturing local keyword clusters rather than just single words. For example, capturing some ngram at the origin of the local maxima would have yielded more information.

Although Yang developed this method further to hierarchially segment text. Following the example of Witkin, the keyword and segmentation methods could be enriched by building keyword trees. [8] This could be done using using a method similar to Yang's, or a hierarchical clustering method could be applied directly to the marignal signal along the spatial axis.

The keywording technique as implemented failed to take into account bigrams (as keywords) but also longer phrases and named entities were not handled. Lebanon formed a high-frequency peak in the text although it appeared once and has few neighbors in the semantic graph. 'Yang', on the other hand, did not appear in the semantic graph but isolated nodes should have been smoothed in a way roughly similar to a node with few neighbors. This difference may be an implementation error or in the handling of disconnected or relatively isolated nodes.

While the results of the initial test on a simple task such as finding keywords was not that impressive, there are aspects of the general approach that still seem promising. In particular, the methods explored so far only touch the surface of 2d signal processing techniques. Additionally, the method lends itself to visualization of larger documents or collections of small documents. Before expanding to new techniques, however, the procedure for semantic filtering would need to be better refined and more robustly tested.

References

[1] DM Blei and JD Lafferty. Dynamic topic models. *Proceedings of the 23rd international conference* . . . , 2006.

[2] KW Church and P Hanks. Word association norms, mutual information, and lexicography. *Computational linguistics*, 1990.

[3] G Lebanon, Y Mao, and JV Dillon. The Locally Weighted Bag of Words Framework for Document Representation. *Journal of Machine Learning* . . . , 2007.

[4] T Lindeberg. Discrete derivative approximations with scale-space properties: A ba-

sis for low-level feature extraction. *Journal of Mathematical Imaging and Vision*, 1993.

[5] T Lindeberg. Scale-space theory: A basic tool for analyzing structures at different scales. *Journal of applied statistics*, 1994.

[6] Q Mei, D Cai, D Zhang, and CX Zhai. Topic modeling with network regularization. ... *of the 17th international conference on* ..., 2008.

[7] G Recchia and MN Jones. More data trumps smarter algorithms: Comparing pointwise mutual information with latent semantic analysis. *Behavior research methods*, 2009.

[8] AP Witkin. Scale-space filtering: A new approach to multi-scale description. *Acoustics, Speech, and Signal Processing, IEEE* ..., 1984.

[9] Shuang-Hong Yang. A Scale-Space Theory for Text. page 9, December 2012.